# SNAP package and Improved QRGCD algorithm (Computer Algebra : The Algorithms, Implementations and the Next Generation)

AUTHOR(S):

増井, 貴明; 長坂, 耕作

CITATION:

増井, 貴明 ...[et al]. SNAP package and Improved QRGCD algorithm (Computer Algebra : The Algorithms, Implementations and the Next Generation). 数理解析研究所講究録 2013, 1843: 101-113

ISSUE DATE:

2013-07

URL:

http://hdl.handle.net/2433/195006

数理解析研究所講究録
第1843巻 2013年 101-113



# SNAP package and Improved QRGCD algorithm

増井貴明
TAKAAKI MASUI*
神戸大学人間発達環境学研究科
GRADUATE SCHOOL OF HUMAN DEVELOPMENT AND ENVIRONMENT, KOBE UNIVERSITY

長坂耕作
KOSAKU NAGASAKA †
神戸大学人間発達環境学研究科
GRADUATE SCHOOL OF HUMAN DEVELOPMENT AND ENVIRONMENT, KOBE UNIVERSITY

## 1 はじめに

数式処理において，多項式の最大公約因子 (GCD) 計算は重要なトピックのひとつであり，多くのアルゴリズムが知られている (詳しくは [4] を参照されたい)．しかしながら，実際の問題 (制御理論や画像処理など) において，多くの場合，入力多項式の係数は浮動小数点数による数値計算の結果や実験値などから表されており，一般的に誤差を含んでいる．このような多項式に対しては通常のアルゴリズムではGCDを計算することは難しい．このような問題に対して**近似GCD計算**という考え方が提唱され，これまでに多くの研究が行われている．

QRGCD [2] (以下，[2] を元論文と呼ぶ) は一変数多項式の近似GCD計算を行うアルゴリズムであり，数式処理ソフトウェア (CAS) Mapleの近似計算ライブラリであるSNAPパッケージにも実装されている．従って多くの新しく提唱されるアルゴリズムに対してベンチマークとしての役割を担っていると言える．しかしながら，QRGCDの理論的な背景はあまり明らかにされておらず，元論文の主定理の証明に誤りがあること，元論文のアルゴリズムとSNAPで実装されているアルゴリズムとの間にはいくつかの相違点があることが分かった．例えば，Bini, Boito [1] らは微小主係数多項式に対してQRGCDは正しい次数の近似GCDを計算することは難しいと述べているが，このことは実際にはSNAPの実装における前処理が関係しているものの元論文などでは明記されていない．本講演では，これらのことについて速報する．

本稿では，2章で元論文のQRGCDの概要について簡単に説明を行い，元論文で挙げられている主定理についての修正を行う．3章では，元論文のアルゴリズムとSNAPで実装されているアルゴリズムとの相違点や問題点について述べ，4章では，修正された主定理に基づく改善版QRGCDアルゴリズム (ExQRGCD) を提案する．5章でこれらのアルゴリズムに対していくつかの数値実験を行う．

なお，本講演で扱う一変数多項式の近似GCDは以下の定義によるものとする．

*masui@main.h.kobe-u.ac.jp
†nagasaka@main.h.kobe-u.ac.jp

**定義 1 (一変数多項式の近似 GCD)**
$f(x), g(x) \in \mathbb{R}[x]$ に対して，$d(x) \in \mathbb{R}[x]$ が摂動の許容度 $\varepsilon > 0$ における $f(x), g(x)$ の**近似 GCD** であるとは，$d(x)$ が次を満たすときである．

$$f(x) + \Delta_f(x) = f_1(x)d(x),\ g(x) + \Delta_g(x) = g_1(x)d(x).$$

ここで，$\Delta_f(x), \Delta_g(x), f_1(x), g_1(x) \in \mathbb{R}[x]$ は

$$\deg(\Delta_f) \le \deg(f), \deg(\Delta_g) \le \deg(g), \|\Delta_f(x)\|_2 < \varepsilon\|f(x)\|_2, \|\Delta_g(x)\|_2 < \varepsilon\|g(x)\|_2$$

を満たすものである ($f_1(x)$, $g_1(x)$ はそれぞれ $f(x)$, $g(x)$ の余因子と呼ぶ)． ◁

# 2 理論的な概要

## 2.1 QRGCD の復習

本章では，QRGCD の元論文 [2] に基づく QRGCD の概要について述べる．次数が $m, n$ $(m \ge n)$ の二つの一変数多項式 $f(x), g(x) \in \mathbb{R}[x]$ を

$$\begin{aligned} f(x) &= f_m x^m + f_{m-1}x^{m-1} + \cdots + f_1 x + f_0, \\ g(x) &= g_n x^n + g_{n-1}x^{n-1} + \cdots + g_1 x + g_0 \end{aligned} \tag{1}$$

とし，$f(x)$ と $g(x)$ はそれぞれ単位 2-ノルムを持つとする (すなわち，$\|f(x)\|_2 = \|g(x)\|_2 = 1$)．このとき，$f(x)$ と $g(x)$ の Sylvester 行列を次で定義する．

**定義 2 (Sylvester 行列)**

$$Syl(f,g) = \begin{bmatrix} f_m & f_{m-1} & \cdots & f_1 & f_0 & & \\ & f_m & f_{m-1} & \cdots & \ddots & \ddots & \\ & & \ddots & \ddots & & \ddots & \ddots \\ & & & f_m & f_{m-1} & \cdots & f_1 & f_0 \\ g_n & g_{n-1} & \cdots & g_1 & g_0 & & & \\ & g_n & g_{n-1} & \cdots & \ddots & \ddots & & \\ & & \ddots & \ddots & & \ddots & \ddots & \\ & & & g_n & g_{n-1} & \cdots & g_1 & g_0 \end{bmatrix}. \tag{2}$$

◁

以下特に断らない限り，ベクトルおよび行列に対して $\|\cdot\|_2$ を 2-ノルム，$\|\cdot\|_F$ をフロベニウスノルムとする．Sylvester 行列の QR 分解と $f(x)$ と $g(x)$ の GCD との関係についてよく知られている定理を以下に示す．

**定理 3**
(2) 式の $QR$ 分解を $Syl(f,g) = QR$ とする．ここで $Q \in \mathbb{R}^{(m+n)\times(m+n)}$ は直交行列，$R \in \mathbb{R}^{(m+n)\times(m+n)}$ は上三角行列である．このとき，$R$ の $\vec{0}$ でない最後の行は $f(x)$ と $g(x)$ の (*exact* な) *GCD* の係数ベクトルである． ◁

数値計算において，QR 分解は後退誤差の面では安定していることが知られているが，前進誤差の面では不安定になることがある．つまり，$S = Syl(f,g)$ に対して数値計算による QR 分解の結果が $\hat{Q}\hat{R}$ だったとき，後退誤差の $\Delta S = \hat{Q}\hat{R} - S$ が十分小さくても前進誤差である $\Delta R = \hat{R} - R$ が小さいことは保証されない．よって元論文では近似 GCD に対応する根の検出に関して，以下の重要な二つの定理が示されている．

**定理 4 (Theorem 2, [2])**
$\omega$ を $f(x) + \Delta_f(x), g(x) + \Delta_g(x)$ の ($\mathbb{C}$ 上の) 単位円内にある共通根とする ( すなわち，$|\omega| < 1$)．このとき，$R$ の $\vec{0}$ でない ( 近似的に $\vec{0}$ でない ) 最後の行は $\omega$ を含む $f(x)$ と $g(x)$ の近似 $GCD$ の近似因子の係数ベクトルである． ◁

**定理 5 (Theorem 3, [2])**
$f(x)$ と $g(x)$ の $Sylvester$ 行列の $QR$ 分解では，$f(x)$ と $g(x)$ の ($\mathbb{C}$ 上の) 単位円外にある共通根を検出することは難しい． ◁

また，$p(x) \in \mathbb{R}[x]$ に対して，**reversal polynomial** $rev(p) \in \mathbb{R}[x]$ を

$$rev(p) = x^{\deg(p)} \cdot p(1/x)$$

で定義する．これらの定理および定義により，アルゴリズムは次のようになる．

**アルゴリズム 1 (QRGCD [2])**

| | |
|---|---|
| **Input** | $f(x), g(x) \in \mathbb{R}[x]$，摂動の許容度 $\varepsilon > 0$ |
| **Output** | $u(x), v(x), d(x) \in \mathbb{R}[x]$[1] ($d(x)$ が $f(x)$ と $g(x)$ の近似 $GCD$) |
| *Step 1* | $Syl(f,g) = QR$ を計算する． |
| *Step 2* | $R$ の右下から $(k+1)\times(k+1)$ 次部分行列を $R_{22}^{(k)}$ とし，$\|R_{22}^{(k)}\|_2 > \varepsilon$ かつ $\|R_{22}^{(k-1)}\|_2 < \varepsilon$ を満たすとする．この $R_{22}^{(k)}$ に対して以下のギャップを満たす $k$ を探す．<br>**(Case 1, 近似的に互いに素):** $\|R_{22}^{(0)}\|_2 > \varepsilon$<br>*(i)* $d_1(x) := 1$，$u(x), v(x)$ を $Q^T$ の一番下の行から計算する．<br>**(Case 2, 大きなギャップ検出):** $\|R_{22}^{(k-1)}\|_2 < 10\varepsilon\|R_{22}^{(k)}\|_2$<br>*(i)* $d_1(x) := R$ の下から $k$ 番目の行から計算する．<br>**(Case 3, ギャップ検出):** $\exists k_1, \|R_{22}^{(k_1-1)}\|_2 < 10\varepsilon\|R_{22}^{(k_1)}\|_2$<br>*(i)* $d_1(x) := R$ の下から $k_1$ 番目の行から計算する．<br>**(Case 4, Difficult case):** それ以外<br>*(i)* $f(x), g(x)$ に対してアルゴリズム *"Split"* を行い，$d_1(x)$ として $f(x), g(x)$ の単位円内にある共通根を計算する． |
| *Step 3* | $f(x)$ と $g(x)$ の $d_1(x)$ に関する余因子の *reversal polynomial* に対して，*Step 1 - 2* を行い，$d_2(x)$ を計算する． |
| *Step 4* | $f(x)$ と $g(x)$ の $d(x) = d_1(x)d_2(x)$ に関する余因子に対して，*Step 1 - 2* を行い，*Bézout* 係数 $u(x), v(x)$ を計算する． |
| *Step 5* | $u(x), v(x), d(x)$ を返す． |

[1] 元論文では $u(x), v(x), d(x)$ の条件として $\|f(x) - f_1(x)d(x)\| < \varepsilon, \|g(x) - g_1(x)d(x)\| < \varepsilon, \|u(x)f(x) + v(x)g(x) - d(x)\| < \varepsilon$ が記されているが，証明も与えられておらず保証されないことから，ここでは省略する．

簡単にまとめると，アルゴリズムの流れは次のようになる．

- 単位円内にある近似 GCD の因子 $d_1(x)$ は，$Syl(f,g)$ の QR 分解により求める (normal side と呼ぶ)．
- 単位円外にある近似 GCD の因子 $d_2(x)$ は，reversal polynomial を用いて $Syl(rev(f/d_1), rev(g/d_1))$ の QR 分解により求める (reversal side と呼ぶ)．
- $f(x)$ と $g(x)$ の近似 GCD として，$d(x) = d_1(x) \cdot d_2(x)$ を求める．

**注意 1 ("Split" の概要)**
*"Split"* は入力多項式に対して，単位円内にある根と単位円外にある根に分離するアルゴリズムであり，*Graeffe's root-squaring, contour integration, Newton's formula, Newton's iteration, lifting steps* に基づいている．*"Split"* の目的については後ほど述べる． ◁

## 2.2 諸定理の修正

前述の定理 4 および定理 5 に関して，元論文で与えられている証明にはいくつかの問題点があることが分かった．以下にその概要を示す．簡単のため，$S = Syl(f,g), S+\Delta_S = Syl(f+\Delta_f, g+\Delta_g)$ とし (すなわち，$\Delta_S = Syl(\Delta_f, \Delta_g)$)，QR 分解 $S = QR, S+\Delta_S = \hat{Q}\hat{R}$ および $N, \hat{N}$ をそれぞれ $S, S+\Delta_S$ の零空間を張るベクトルを列に持つ行列とする．また，$d(x)$ を単位 2-ノルムを持つ次数 $k$ の $f(x)$ と $g(x)$ の近似 GCD (すなわち，$f(x)+\Delta_f(x), g+\Delta_g(x)$ の exact な GCD) とする．このとき，

$$SN = (S+\Delta_S)\hat{N} = \vec{0},$$

$$(S+\Delta_S)\hat{N} = (QR+\Delta_S)\hat{N} \Longrightarrow -R\hat{N} = Q^T\Delta_S\hat{N} \tag{3}$$

が成り立つ．

**定理 4 の証明 ([2])** $r(x)$ を $R$ の行 $\vec{r}$ を係数にもつ多項式とし，$\omega$ を $|\omega|<1$ であるような $d(x)$ の根とするとき，ある $N_k \subset \hat{N}$ があって，$|r(\omega)| \leq \|\Delta S\|_2 \|N_k\|_2$ が成り立つことから $d(x)$ と (2-ノルムの意味で) 近い $r(x)$ が検出できるという流れになっている．しかしながら，これは $r(x)$ の根 $\omega$ における絶対評価であり，$d(x)$ と $r(x)$ が係数ベースで相対的に十分近いことは言っていない (例えば，$10^{-16}(x-0.5)(x-0.8)$ と $10^{-16}(x+0.5)(x+0.8)$)．

**定理 5 の証明 ([2])** $\vec{\omega}_* = (\omega^k, \cdots, \omega^1, \omega^0)^T \in N_k$ で (3) 式に対して $(S+\Delta_S)\vec{\omega}_* = \hat{Q}\hat{R}\vec{\omega}_* = \vec{0}$ が成り立つことから，$S$ の QR 分解で近似 GCD を検出するためには，$\Delta_S\vec{\omega}_*$ のノルムが十分小さくなる必要がある．よってもし単位円外にある共通根 ($|\omega|>1$) であれば近似 GCD の検出は難しいという流れになっている．この議論は近似 GCD を根ベースで考える場合には正しいが，本稿および元論文の係数ベースで考える場合には根がずれてしまう可能性があるため正しいとは言えない．

以上の理由から，本章では定理 4 および定理 5 に類似した形で新しい定理を修正版として示すこととする．そのために，いくつかの補題を以下に示す．

**補題 6**
$S = QR, S+\Delta_S = \hat{Q}\hat{R}$ とし，$\hat{N}$ を $S+\Delta_S$ の零空間を張るベクトルを列に持つ行列とする．このとき，$\forall\vec{\omega}_* \in \hat{N}$ に対して $|r(\omega)| \leq \|\Delta_S\|_2\|\vec{\omega}_*\|_2$ が成り立つ． ◁

**証明** (3) 式より，

$$-R\vec{\omega}_* = Q^T\Delta_S\vec{\omega}_*.$$

したがって，

$$\begin{aligned}
&\Longrightarrow \|R\vec{\omega}_*\|_2 = \|Q^T\Delta_S\vec{\omega}_*\|_2 \\
&\Longrightarrow \|R\vec{\omega}_*\|_2 \leq \|\Delta_S\vec{\omega}_*\|_2 \\
&\Longrightarrow \|r(\omega)| \leq \|R\vec{\omega}_*\|_2 \leq \|\Delta_S\vec{\omega}_*\|_2 \leq \|\Delta_S\|_2\|\vec{\omega}_*\|_2.
\end{aligned}$$

∎

また，次の補題に関する準備として，$p(x) = \sum_{i=0}^{k} p_i x^i$, $p_i \in \mathbb{C}$ に対して，多項式集合 $\mathcal{P}_\varepsilon(p)$ を

$$\mathcal{P}_\varepsilon(p) = \left\{ \tilde{p}(x) \in \mathbb{C}[x] \;\middle|\; \|\vec{p} - \vec{\tilde{p}}\|_2 \leq \varepsilon\|\vec{p}\|_2,\ \deg(\tilde{p}) \leq \deg(p) \right\}$$

で定義する．ここで，$\vec{p} = (p_k, \ldots, p_0)$, $\vec{\tilde{p}} = (\tilde{p}_k, \ldots, \tilde{p}_0)$ である．

**補題 7 (Corollary 7, [5])**
$\omega \in \mathbb{C}$ に対して[2]，

$$\exists \tilde{p}(x) \in \mathcal{P}_\varepsilon(p),\ \tilde{p}(\omega) = 0 \Longleftrightarrow |p(\omega)| \leq \varepsilon\|p(x)\|_2\|\vec{\omega}_*\|_2.$$

◁

これらの補題により前述の定理 4 を次のように修正する．

**定理 8 (修正版 定理 4)**
これまでの記法に従うものとし，$r(x)$ を $R$ の下から $(k+1)$ 番目の行 $\vec{r}$ を係数に持つ多項式とする．このとき，$\frac{\|\Delta_S\|_2}{\|r(x)\|_2} \ll 1$ ならば，$r(x)$ は $f(x)$ と $g(x)$ の近似 $GCD$ である． ◁

**証明** 補題 6 より，$|r(\omega)| \leq \|\Delta_S\|_2\|\vec{\omega}_*\|_2 = \frac{\|\Delta_S\|_2}{\|r(x)\|_2} \cdot \|r(x)\|_2\|\|\vec{\omega}_*\|_2$ が成り立つ．もし $\frac{\|\Delta_S\|_2}{\|r(x)\|_2} = \varepsilon \ll 1$ ならば，$|r(\omega)| \leq \varepsilon\|r(x)\|_2\|\vec{\omega}_*\|$ である．よって補題 7 から，$d(x)$ を $f(x) + \Delta_f(x), g(x) + \Delta_g(x)$ の exact な GCD とすると，$\|\vec{r} - \vec{d}\|_2 \leq \varepsilon\|\vec{r}\|_2$ が成り立つ．よって $r(x)$ は $f(x)$ と $g(x)$ の近似 GCD である． ∎

次に，単位円外にある共通根の検出性に関して，次の補題を述べる．

**補題 9**
*Sylvester* 行列の $QR$ 分解により得られた $R$ に対して，$r(x)$ を $R$ のある行 $\vec{r}$ を係数に持つ多項式とする．このとき，$r(x)$ に含まれる根は相対的に近いものから検出されるわけではない．

**証明** 証明については，本稿では省略するが，あらすじとしては Sylvester's single sum [3] に基づいて $r(x)$ に含まれる根の大きさと PRS の関係について調べることができる．発表では割愛したものとして，以下に具体例を挙げる． ∎

**例 1 (artificial common inside roots)**
次の $f(x)$ と $g(x)$ に対して *Sylvester* 行列の $QR$ 分解を考える．

$$\begin{aligned}
f(x) &= (x + 0.01)(x - 0.01)(x + 100)(x - 100) \\
g(x) &= (x + 0.010000001)(x - 0.09)(x + 100.00001)(x - 200)
\end{aligned}$$

ここで，摂動の許容度 $\varepsilon = 10^{-12}$ に対して $f(x)$ と $g(x)$ の近似 $GCD$ $d(x)$ は，

$$\begin{aligned}
d(x) &= 9.99998 \times 10^{-4}x^2 + 0.999999x + 9.99998 \times 10^{-4} \\
&\approx 9.99998 \times 10^{-4}(x + 0.00100000)(x + 1000.00)
\end{aligned}$$

---

[2] [5] では双対ノルムについて示されているが，ここでは 2-ノルムの相対評価に書き換えている．

となる．しかしながら，$R$ の下から 2 番目，3 番目および 4 番目の行ベクトルを係数に持つ多項式は次のようになる．

$$\begin{aligned}
4\text{番目: } & 0.00143003x^3 + 1.39855x^2 - 0.00559397x - 6.99252 \times 10^{-6} \\
& \approx 0.00143003(x - 0.00499981)(x + 0.001)(x + 977.993), \\
3\text{番目: } & 0.20978x^2 - 0.000839074x - 1.04885 \times 10^{-6} \\
& \approx 0.20978(x - 0.00499978)(x + 0.001), \\
2\text{番目: } & 0.00565685x + 5.65685 \times 10^{-6} \\
& \approx 0.00565685(x + 0.001).
\end{aligned}$$

3 番目の多項式は $d(x)$ とは無関係な $f(x)$ と $g(x)$ の共通根でないものを含んでいる．この影響により本来の単位円外にある共通根が正しく検出されていない．以後このような単位円外にある共通根のかわりに検出されてしまう単位円内の根を **artificial common inside roots** と呼ぶ． ◁

以上の理由から前述の定理 5 を次のように修正する．

**定理 10 (修正版 定理 5)**
*Sylvester* 行列の *QR* 分解では，単位円外の共通根を含む近似 *GCD* を計算することは難しい．正しく計算を行うには，多くの場合に *normal side* と *reversal side* の両方から計算を行い，それらを合わせる必要がある． ◁

# 3 SNAP の実装

QRGCD は Maple の SNAP パッケージとして実装されている．本章では，まず SNAP の実装を示し (アルゴリズム 2)，元論文のアルゴリズムとの間にあるいくつかの相違点について述べる．なお，行列 $A$ に対して $\|A\|_S$ は $A$ の各行ベクトルの 2-ノルムの和を表すとする (すなわち，$\|A\|_S = \sum \|\vec{a}_i\|_2, A = (\vec{a}_1, \ldots, \vec{a}_n)^T$)．

**注意 2 ("Split" の目的)**
元論文でも用いられているアルゴリズム *"Split"* の単位円内の根と単位円外の根に分離する目的に関しては元論文では明記されていない．分離する理由として考えられることは次の二つである，一つ目は低次の多項式に対する *QR* 分解の方が高次のものに比べて前進誤差がより小さくなることで精度が保たれることにある．二つ目は前述の *artificial common inside roots* の影響により，単位円外にある共通根が正しく検出できない可能性があることである． ◁

## 3.1 元論文との相違点

Maple の SNAP における実装が元論文のアルゴリズムと異なる点として，以下の 4 点について述べる．

**用いるノルム** SNAP の実装では元論文で用いられている 2-ノルムではなく，$\|\cdot\|_S$ が用いられているが，そのことは元論文では明記されていない．任意の行列 $A$ に対して，$\|A\|_2 \leq \|A\|_F \leq \|A\|_S$ であることが分かるが，$\|A\|_S$ は緩和しすぎではないかと考えられる．実験により計算される近似 GCD に大きな違いが見受けられないことは確認したが，後述の改善版 QRGCD では計算コストとの兼ね合いからもフロベニウスノルムを採用することとする．

**アルゴリズム 2 (QRGCD in the SNAP package)**

| | |
|---|---|
| **Input** | $f(x), g(x) \in \mathbb{R}[x]$, 摂動の許容度 $\varepsilon > 0$ |
| **Output** | $u(x), v(x), d(x) \in \mathbb{R}[x]$ *such that* $\|f(x) - f_1(x)d(x)\|_2 + \|g(x) - g_1(x)d(x)\|_2 < 10\varepsilon$, *or "failure"* |
| *Step 1* | 閾値を $\varepsilon/100$ として *"find non-zero terms"* を $f(x), g(x)$ に対して行う. |
| *Step 2* | $Syl(f,g) = QR$ を計算する. |
| *Step 3* | $R$ の右下から $(k+1)\times(k+1)$ 次部分行列を $R_{22}^{(k)}$ とし, $\|R_{22}^{(k)}\|_S > \varepsilon$ かつ $\|R_{22}^{(k-1)}\|_S < \varepsilon$ を満たすとする. この $R_{22}^{(k)}$ に対して以下のギャップを満たす $k$ を探す.<br>**(Case 1, 近似的に互いに素):** $\|R_{22}^{(0)}\|_S > \varepsilon$<br>*(i)* $d_1(x) := 1$, $u(x), v(x)$ を $Q^T$ の一番下の行から計算する.<br>**(Case 2, 大きなギャップ検出):** $\|R_{22}^{(k-1)}\|_S < 10\varepsilon\|R_{22}^{(k)}\|_S$<br>*(i)* $d_1(x) :=$ $R$ の下から $k$ 番目の行から計算する.<br>**(Case 3, ギャップ検出):** $\exists k_1, \|R_{22}^{(k_1-1)}\|_S < 10\varepsilon\|R_{22}^{(k_1)}\|_S$<br>*(i)* $d_1(x) :=$ $R$ の下から $k_1$ 番目の行から計算する.<br>**(Case 4, Difficult case):** それ以外<br>*(i)* $R$ の行ベクトルで 2-ノルムが $5000\varepsilon$ 以上であるものを係数に持つ多項式を $p_1(x), p_2(x), (\deg(p_1) > \deg(p_2))$ とする.<br>*(ii)* アルゴリズム *"Split"* を用いて $p_1(x)$ の単位円内にある根からなる多項式 $p_{1_{in}}(x)$ を計算する.<br>*(iii)* $p_{1_{in}}(x)$ と $p_2(x)$ に対して *Step 2 - 3* を行い, $d_1(x)$ を計算する. |
| *Step 4* | $d_1(x) = 1$ ならば $rev(f), rev(g)$ に対して再帰呼び出しを行い, 結果の *reversal polynomial* をとる. |
| *Step 5* | $f(x), g(x)$ の $d_1(x)$ に関する余因子の *reversal polynomial* に対して *Step 2 - 3* を行い, $d_2(x)$ を計算する. |
| *Step 6* | $f(x), g(x)$ の $d(x) = d_1(x)d_2(x)$ に関する余因子に対して *Step 2 - 3* を行い, $u(x), v(x)$ を計算する. |
| *Step 7* | $u(x), v(x), d(x)$ を返す. |

**"Split" を適用する多項式** 元論文では入力多項式に対して "Split" を適用するが, SNAP の実装では, 上三角行列 $R$ に現れる多項式 (すなわち, PRS) の一つに対してのみ "Split" を適用する. PRS の Split は入力多項式の Split に比べて次数が低いので, 計算コストが小さくて済むと考えられるが, $5000\varepsilon$ 以上の 2-ノルムを持つ多項式を選ぶ根拠については示されておらず, 実験的な値であると考えられる.

**fail-safe ループ** 元論文のアルゴリズムでは, 常に単位円内にある共通根から検出を試み, その後で単位円外にある共通根の検出を行う. しかし, もし単位円外にある共通根の個数が単位円内のものより非常に多い場合, このような単位円内にある共通根は検出が難しい可能性があるため, 単位円外にある共通根から検出を試みる方がよい. SNAP では normal side から計算を行って, 近似 GCD が計算されない場合に reversal side から計算し直す実装になっている.

ここでは, 特に重要な相違点として, SNAP の前処理の一つである "find non-zero terms" について以下で詳しく述べることとする.

**"find non-zero terms"** SNAP の実装では前処理の一つとして "find non-zero terms" が実行される. このアルゴリズムは, 入力多項式に対して主係数および定数を $\varepsilon/100$ 以上となるように書き換えるもの

である．たとえば，$f(x)=\sum_{i=0}^{m} f_i x^i$ に対して，

$$|f_m|,|f_{m-1}|,\ldots,|f_{m'+1}|<\varepsilon/100,|f_{m'}|\geq\varepsilon/100,$$
$$|f_{m''}|\geq\varepsilon/100,|f_{m''-1}|,\ldots,|f_0|<\varepsilon/100$$

とすると，"find non-zero terms" によって $f(x)=\sum_{i=0}^{m'-m''} f_{m''+i}x^i$ となる．この前処理によって微小主係数多項式に対して QRGCD を計算すると，いくつかの問題が生じることが分かった．Bini, Boito [1] らは微小主係数多項式が QRGCD の ill-conditioned であると述べているが，このことは次の実験から，"find non-zero terms" を除いた SNAP の実装と比較することで微小主係数多項式が QRGCD の ill-conditioned ではないことが分かる．

**例 2 (微小主係数多項式)**
この実験は *Bini, Boito [1]* により行われており，計算される近似 *GCD* の次数が問題となる．ここでは *10000* ペアの以下の多項式を生成した．

$$f(x)=(\alpha x^3+2x^2-x+5)(x^4+7x^2-x+1),$$
$$g(x)=(\alpha x^3+2x^2-x+5)(x^3-x^2+4x-2).$$

上で，$\alpha\in[10^{-10},10^{-5}]$ はランダムに選ぶ．これら $f(x),g(x)$ に対して *Maple 16* 上で摂動の許容度 $\varepsilon=10^{-5}$ で実験を行った (*Digits* := 16)．表 *1* で，*"ExQRGCD"* , *"SNAP"* および *"SNAP (w/o f.n.t)"* はそれぞれ我々の改善版 *QRGCD* アルゴリズム，*SNAP* の実装，*"find non-zero terms"* を除いた *SNAP* の実装を表す．結果として，*"find non-zero terms"* の影響により近似 *GCD* として正しい次数のものが検出されていないことが分かる．一方で，*SNAP (w/o f.n.t)* と *ExQRGCD* は全ての多項式に対して次数 *3* のものを検出できている． ◁

| | #(deg = 3) | #(deg = 2) | $\|\Delta_f(x)\|_2$ | $\|\Delta_g(x)\|_2$ |
|---|---|---|---|---|
| ExQRGCD | 10000 | 0 | $3.64\times10^{-14}$ | $2.18\times10^{-14}$ |
| SNAP | 742 | 9254 | $2.04\times10^{-6}$ | $6.06\times10^{-7}$ |
| SNAP (w/o f.n.t) | 10000 | 0 | $8.15\times10^{-14}$ | $3.84\times10^{-14}$ |

表 1: 微小主係数多項式：実験結果

# 4 改善版 QRGCD アルゴリズム

本章では，2.2 章で述べた修正版定理に基づいた，改善版 QRGCD アルゴリズム (ExQRGCD) について述べる．ExQRGCD (アルゴリズム 3) は SNAP や元論文より計算速度は遅いが，数学的根拠に基づいており，与えた摂動の許容度 $\varepsilon$ を満たすことを保証している．ExQRGCD では，行列のノルム計算に関して計算コストを抑えるために，2-ノルムのかわりにフロベニウスノルムを採用している．また，artificial common inside roots の影響を抑えるために ExQRGCD では，理想的な近似 GCD に十分近いものから少しずつ計算を行う．従って，normal side と reversal side の計算を何度も繰り返す必要があり，計算コストは元論文や SNAP に比べると多くかかる．以下では元論文を書き換えた Case 1, 2 について詳しく述べることとする．

**アルゴリズム 3 (ExQRGCD)**

| | |
|---|---|
| **Input** | $f(x), g(x) \in \mathbb{R}[x]$, 摂動の許容度 $\varepsilon > 0$ |
| **Output** | $u(x), v(x), d(x) \in \mathbb{R}[x]$ *such that* $\|f(x) - f_1(x)d(x)\|_2 < \varepsilon, \|g(x) - g_1(x)d(x)\|_2 < \varepsilon$ |
| *Step 1* | *first side* (*normal* か *reversal*) を決定する. $i := 1$, <br> *first side* が *normal* $\Longrightarrow f_1(x) := f(x), g_1(x) := g(x)$, <br> *first side* が *reversal* $\Longrightarrow f_1(x) := rev(f), g_1(x) := rev(g)$ |
| *Step 2* | $Syl(f,g) = QR$ を計算する. |
| *Step 3* | $R$ の右下から $(k+1) \times (k+1)$ 次部分行列を $R_{22}^{(k)}$ とし, $\vec{r}_k$ を $R$ の下から $(k+1)$ 番目の行ベクトルとする. <br> **(Case 1, 近似的に互いに素):** $\|R_{22}^{(0)}\|_F > \varepsilon\sqrt{m+n}$ <br> *(i)* $d_i(x) := 1$. <br> **(Case 2, 試し割り):** $\|R_{22}^{(0)}\|_F \leq \varepsilon\sqrt{m+n}$ <br> *(i)* $\|R_{22}^{(k-1)}\|_F \leq \varepsilon\sqrt{m+n}$ を満たす全ての $k \geq 1$ に対して, $er_k := \|R_{22}^{(k-1)}\|_F / \|\vec{r}_k\|_2$ を計算する. <br> *(ii)* 三つの最小の $er_k$ に対して, <br> $d_i(x) := r_k(x)$ とし, $d(x) := \prod d_i(x)$ に対して $\exists f_1(x), g_1(x)$ *such that* $\|f(x) - f_1(x)d(x)\|_2 < \varepsilon$ かつ $\|g(x) - g_1(x)d(x)\|_2$ ならば, *Step 4* へ進む ( 以下の *(v)* における再帰呼び出しにおいて, $f_1(x), g_1(x)$ が見つからなければ $d_i(x) := 1$ として *Step 4* へ進む ). <br> *(iii)* $R$ の行ベクトルで *2-*ノルムが *1.0* 以上となる *2* つのものをとり, それぞれ $p_1(x), p_2(x) (\deg(p_1) > \deg(p_2))$ とする. <br> *(iv)* $p_1(x), p_2(x)$ に対してアルゴリズム *"Split"* を行い, 単位円内にある根のみからなる多項式 $p_{1_{\text{in}}}(x), p_{2_{\text{in}}}(x)$ を求める. <br> *(v)* $p_{1_{\text{in}}}(x), p_{2_{\text{in}}}(x)$ に対して *Step 2 - 3* を行い, $d_i(x)$ を計算する. |
| *Step 4* | $i := i + 1$ とし, *normal* $\leftrightarrow$ *reversal* を入れ替え, $d(x) := \prod d_i(x)$ に関する余因子 $f_1(x), g_1(x)$ (*reversal side* ならば *reversal polynomial* をとったもの ) に対して *Step 2 - 3* を行い, ある $i$ に対して $d_i(x) = d_{i+1}(x) = 1$ となるまで, $d_{i+1}(x)$ を計算する. |
| *Step 5* | $f_1(x), g_1(x), d(x)$ を返す. |

## 4.1 Case 1 の判定：近似的に互いに素

QRGCD と ExQRGCD は, 与えられた二つの多項式 $f(x), g(x)$ の Sylvester 行列 $S = Syl(f,g)$ を QR 分解することで $R$ の行ベクトルを係数に持つ多項式として近似因子を検出するアルゴリズムであることは既に述べたとおりである. 従って, $S + \Delta_S = \hat{Q}\hat{R}$ としたとき $\hat{R}$ から得られる $f(x) + \Delta_f(x), g(x) + \Delta_g(x)$ の exact な GCD である $d(x)$ に対して, QRGCD と ExQRGCD は $S = QR$ の $R$ から得られる $r(x)$ がどれだけ $d(x)$ に近いかという観点に立つものである. よって $d(x)$ そのものを $R$ から直接的に計算することはできない.

ここで, $S = QR$ により $R$ から得られた次数 $k$ の $r(x)$ を exact な GCD として持つ多項式を $f(x) + \delta_f(x)$, $g(x) + \delta_g(x)$ とし, その Sylvester 行列の QR 分解 $S + \delta_S = \tilde{Q}\tilde{R}$ を考える. このとき, Sylvester 行列の QR 分解の性質から

$$S + U,\ \|U\|_F = \|R_{22}^{(k-1)}\|_F \Longrightarrow rank(S+U) = m + n - k$$

であることが分かる．ここで $R_{22}^{(k-1)}$ は unstructured である．一方，本来 $S$ に加えるべき $U$ は structured である必要があるが，多くの場合に structured な行列のノルムは unstructured な行列のノルムより大きいため，

$$\|R_{22}^{(k-1)}\|_F \leq \|\delta_S\|_F$$

が成り立つ．従って，$\|R_{22}^{(k-1)}\|_F > \varepsilon\sqrt{m+n}$ であれば，多くの場合に $\|\delta_S\|_F > \varepsilon\sqrt{m+n}$ が成り立つので，近似的に素である，と判定できる．

### 4.2 Case 2 の判定：試し割り

これまでの議論により，一度の QR 分解では全ての単位円内にある共通根を検出できない可能性がある．ExQRGCD では，ギャップの判定条件として $er_k := \|R_{22}^{(k-1)}\|_F/\|\vec{r}_k\|_2$ を計算するが，$r(x)$ としてできる限り高次数のものを検出するために，artificial common inside roots の影響を最小限にする必要がある．従って，できる限り小さな $er_k$ に関して $r(x)$ を計算しようと考えた．実験により，多くの場合に最小の $er_k$ で摂動の許容度 $\varepsilon$ を十分に満たす近似 GCD の因子が計算できることを確認したが，念のために最小のもの三つを計算することとした．また，"Split" を適用する多項式に対しては，次のように変更した．

**"Split" を適用する多項式** $p_1(x), p_2(x)$ を $f(x), g(x)$ の PRS，$r(x)$ を近似 GCD とする．このとき，

$$f(x) + \delta_f(x) = f_1(x)r(x), g(x) + \delta_g(x) = g_1(x)r(x),$$
$$u_1(x)f(x) + v_1(x)g(x) = p_1(x), u_2(x)f(x) + v_2g(x) = p_2(x)$$

が成り立つ．ここで，$u_1(x), v_1(x), u_2(x), v_2(x)$ は単位 2-ノルムを持ち，$Syl(f,g)$ の QR 分解で得られた直行行列 $Q$ のある列ベクトルを係数にもつ多項式である．ここで，

$$p_1(x) = (u_1(x)f_1(x) + v_1(x)g_1(x))r(x) - (u_1(x)\delta_f(x) + v_1(x)\delta_g(x)),$$
$$p_2(x) = (u_2(x)f_1(x) + v_2(x)g_1(x))r(x) - (u_2(x)\delta_f(x) + v_2(x)\delta_g(x))$$

より，$r(x)$ は摂動の許容度 $(\|\delta_f(x)\|_2 + \|\delta_g(x)\|_2)/\min\{\|p_1(x)\|_2, \|p_2(x)\|_2\}$ における $p_1(x), p_2(x)$ の近似 GCD である．従って，"Split" を適用する多項式として $p_1(x), p_2(x)$ の 2-ノルムが十分小さくない (例えば，$\|p_1(x)\|_2, \|p_2(x)\|_2 \geq 1.0$) ものを取ることとする (このような $p_1(x), p_2(x)$ は $f(x), g(x)$ の共通根以外のものを含んでいる可能性がある)．

## 5 数値実験

以下に，3 つの実験に関してその結果を示す．なお，表記として，ExQRGCD は改善版 QRGCD アルゴリズム，SNAP は SNAP パッケージに実装されている QRGCD アルゴリズム，QRGCD は元論文のアルゴリズムを示している．図 2 において摂動量の平均に関して，縦軸が $log_{10}$ の対数グラフになっていることに注意されたい．なお，発表においては実験 1, 3 のみを扱い，本稿では新たに実験 2 を追加した．

### 実験 1. (exact な根を持つランダム多項式)

$i = 1, \ldots, 10$ に対して，以下を満たす 100 個の多項式ペア $f(x), g(x)$ を生成する．

$$f(x) = f_1(x)d(x), g(x) = g_1(x)d(x),$$
$$d(x) = \sum_{j=0}^{5i} d_j x^j, f_1(x) = \sum_{j=0}^{5i} f_{1,j}x^j, g_1(x) = \sum_{j=0}^{5i} g_{1,j}x^j.$$

上で，$f_{1,j}, g_{1,j}, d_j \in [-99, 99] \subset \mathbb{Z}$ を満たすものをランダムに選び，$f(x), g(x)$ は単位2-ノルム ($\|f(x)\|_2 = \|g(x)\|_2 = 1$) を持つように正規化を行い，$Digits := 10$ で丸めた．これらに対して摂動の許容度 $\varepsilon = 10^{-5}$ で計算した ($Digits := 16$)．

図1より，検出次数に関してExQRGCDとSNAPの間には大きな違いはないが，QRGCDは他に比べて良くない．SNAPとQRGCDは表2から "failure" や与えた摂動の許容度 $\varepsilon$ を満たさないものがいくつか検出されている (ExQRGCDは全ての多項式に対してこのような問題は発生していない)．計算速度に関しては，ExQRGCDとQRGCDはSNAPと比べてそれぞれ，1.59倍，2.08倍遅い結果となった．"failure" の数を除けば，摂動量に関してはSNAPが最も良い結果となった．

## 実験 2. (摂動多項式を加えたランダム多項式)

$i = 1, \ldots, 10$ に対して，以下を満たす100個の多項式ペア $f(x), g(x)$ を生成する．

$$f(x) = f_1(x)d(x)/\|f_1(x)d(x)\|_2 + 10^{-8}\Delta_f(x)/\|\Delta_f(x)\|_2,$$
$$g(x) = g_1(x)d(x)/\|g_1(x)d(x)\|_2 + 10^{-8}\Delta_g(x)/\|\Delta_g(x)\|_2,$$
$$\Delta_f(x) = \sum_{j=0}^{10i} \Delta_{f_j} x^j, \Delta_g(x) = \sum_{j=0}^{10i} \Delta_{g_j} x^j.$$

上で，$\Delta_{f_j}, \Delta_{g_j} \in [-99, 99] \subset \mathbb{Z}$ を満たすものをランダムに選び，$f_1(x), g_1(x), d(x)$ は実験1.の多項式とし，$Digits := 10$ で丸めた．これらに対して摂動の許容度 $\varepsilon = 10^{-5}$ で計算した ($Digits := 16$)．

図1と表2より，ExQRGCDがSNAPとQRGCDに比べて非常に良いことが分かる．計算速度に関しては，ExQRGCDとQRGCDはSNAPと比べてそれぞれ，1.99倍，2.26倍遅い結果となった．"failure" の数を除けば，摂動量に関してはSNAPが最も良い結果となった．

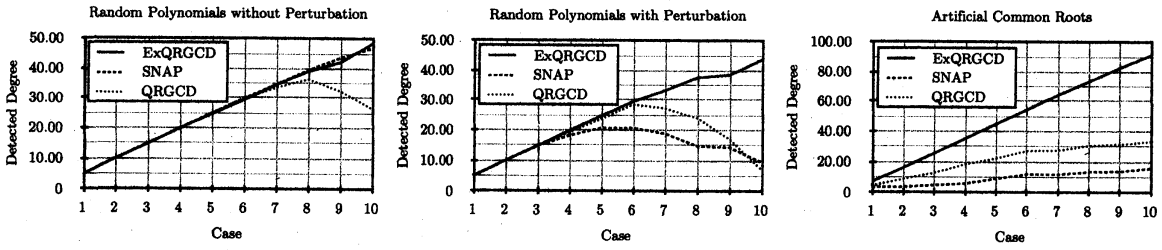


図 1: 検出次数：実験結果 ("failure" は0として換算)

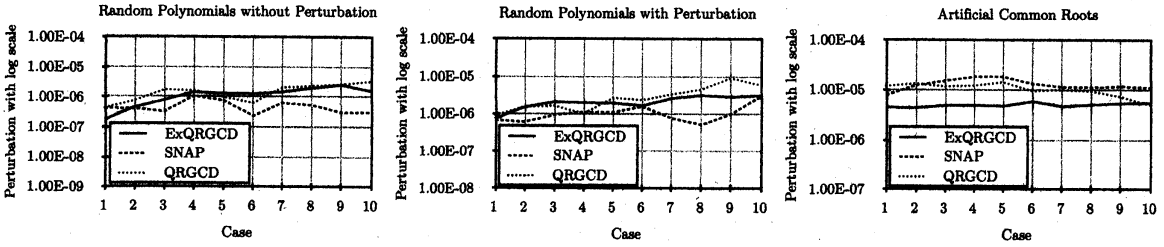


図 2: 摂動量 (すなわち，$(\|\Delta_f\|_2 + \|\Delta_g\|_2)/2$)：実験結果 ("failure" は除く)

| | Random polynomials without Perturbation | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k$=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | $\Delta_f$ | $\Delta_g$ |
| SNAP | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 3 | 6 | 6 | 6 |
| QRGCD | 0 | 0 | 0 | 1 | 0 | 0 | 4 | 9 | 28 | 47 | 26 | 29 |

| | Random polynomials with Perturbation | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k$=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | $\Delta_f$ | $\Delta_g$ |
| SNAP | 0 | 0 | 0 | 9 | 17 | 31 | 46 | 63 | 68 | 81 | 6 | 6 |
| QRGCD | 0 | 0 | 0 | 3 | 5 | 5 | 22 | 40 | 62 | 86 | 31 | 31 |

| | Artificial Common Roots | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $k$=1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | $\Delta_f$ | $\Delta_g$ |
| SNAP | 29 | 63 | 68 | 71 | 64 | 60 | 66 | 66 | 69 | 68 | 110 | 119 |
| QRGCD | 11 | 12 | 14 | 8 | 11 | 9 | 21 | 23 | 30 | 33 | 210 | 216 |

表 2: SNAP, QRGCD における "failure" と適切でない摂動量の数：実験結果

**実験 3. (artificial common roots を含む多項式)**

$i = 1, \ldots, 10$ に対して，以下を満たす 100 個の多項式ペア $f(x), g(x)$ を生成する．

$$f(x) = d(x)\prod_{j=1}^{2i}(x-\omega_{f,j})\prod_{j=1}^{2i}(x-\hat{\omega}_{f,j}),$$
$$g(x) = d(x)\prod_{j=1}^{2i}(x-\omega_{g,j})\prod_{j=1}^{2i}(x-\hat{\omega}_{g,j}),$$
$$d(x) = \prod_{j=1}^{3i}(x-\omega_{d,j})\prod_{j=1}^{3i}(x-\hat{\omega}_{d,j}).$$

上で，$\omega_{\cdot,j} = O(10^{-2}), \hat{\omega}_{\cdot,j} = O(10^2)$ を満たすものをランダムに選び，$f(x), g(x)$ は単位2-ノルム ($\|f(x)\|_2 = \|g(x)\|_2 = 1$) を持つように正規化を行い，$Digits := 10$ で丸めた．これらに対して摂動の許容度 $\varepsilon = 10^{-5}$ で計算した ($Digits := 16$)．

図 1 と表 2 より，ExQRGCD が SNAP と QRGCD に比べて非常に良いことが分かる．計算速度に関しては，ExQRGCD と QRGCD は SNAP と比べてそれぞれ，39.8 倍, 52.6 倍遅い結果となった (SNAP は実験で用いた多項式のうち，約 62%に対して "failure" を返しており，SNAP の計算速度は "failure" の出ない簡単な問題に対してのみの平均となっている)．摂動量に関しては，ExQRGCD が最も良い結果となった．

## 6 まとめ

本講演では，QRGCD アルゴリズムの元論文に関して，主定理の修正を行い，また SNAP の実装と元論文のアルゴリズムの間にあるいくつかの相違点について速報した．特に SNAP の前処理の一つである "find non-zero terms" の影響により，本来 QRGCD の ill-conditioned ではない微小主係数多項式に対して，SNAP が正しく計算できないことを確認した．我々は，修正した主定理に基づき改善版 QRGCD アルゴリ

ズム (ExQRGCD) を提案した．実験結果からは，摂動を含む多項式および artificial common roots を含む多項式に対しては，ExQRGCD が SNAP や QRGCD と比べて非常に良い結果となった．特に artificial common roots を含む多項式に対して，SNAP が多くの場合 "failure" を返す所を，ExQRGCD では与えた摂動の許容度を満たすもので高次数の近似 GCD を計算できていることを確認した．今後は，Modern なアルゴリズム (Fastgcd, UVGCD, GPGCD など) とも比較を行いたい．

## Acknowledgement

This work was supported in part by Japanese Ministry of Education, Culture, Sports, Science and Technology under Grant-in-Aid for Young Scientists, NEXT KAKENHI (22700011).

## 参 考 文 献

[1] Dario A. Bini and Paola Boito. A fast algorithm for approximate polynomial GCD based on structured matrix computations. In *Numerical methods for structured matrices and applications*, Vol. 199 of *Oper. Theory Adv. Appl.*, pp. 155–173. Birkhäuser Verlag, Basel, 2010.

[2] Robert M. Corless, Stephen M. Watt, and Lihong Zhi. $QR$ factoring to compute the GCD of univariate approximate polynomials. *IEEE Trans. Signal Process.*, Vol. 52, No. 12, pp. 3394–3402, 2004.

[3] Carlos D'Andrea, Hoon Hong, Teresa Krick, and Agnes Szanto. An elementary proof of sylvester's double sums for subresultants. *J. Symb. Comput.*, Vol. 42, No. 3, pp. 290–297, March 2007.

[4] Joachim Von Zur Gathen and Jurgen Gerhard. Modern Computer Algebra. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[5] Hans J. Stetter. The nearest polynomial with a given zero, and similar problems. *SIGSAM Bull.*, Vol. 33, No. 4, pp. 2–4, December 1999.